COMP2211

University of Southampton

Software Engineering Group Project

Deliverable 3

Increment 2

Version History

No.	Date	Comments
1	2nd April 2023	First Submission

Group 1 Members

Name	Email
Gavin Teo Siu Chyi	gsct1c21@soton.ac.uk
Yohith V. Nyanasegaran	yvn1e20@soton.ac.uk
Lim Shi Xun	sxl1c21@soton.ac.uk
Ang Jing Ru	jra1c21@soton.ac.uk

Table of Contents

1 Application

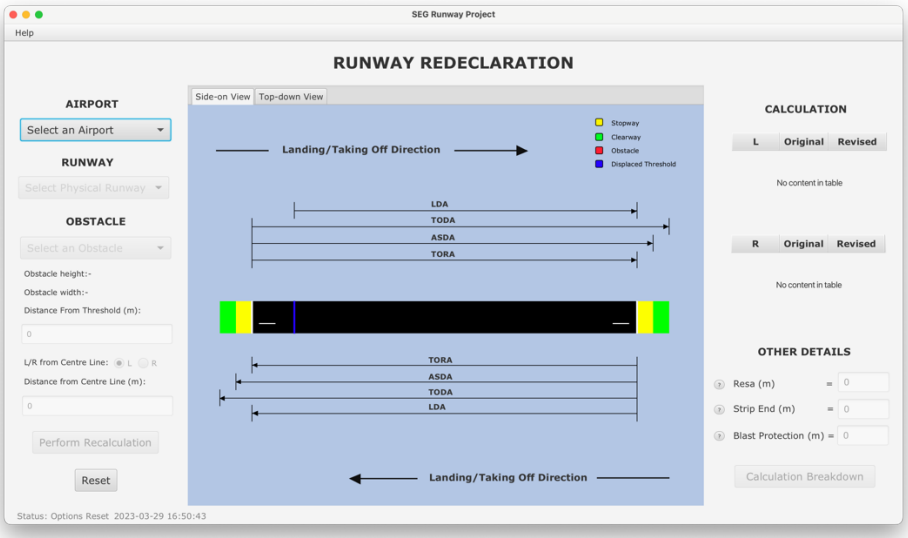
1.1 Product Value

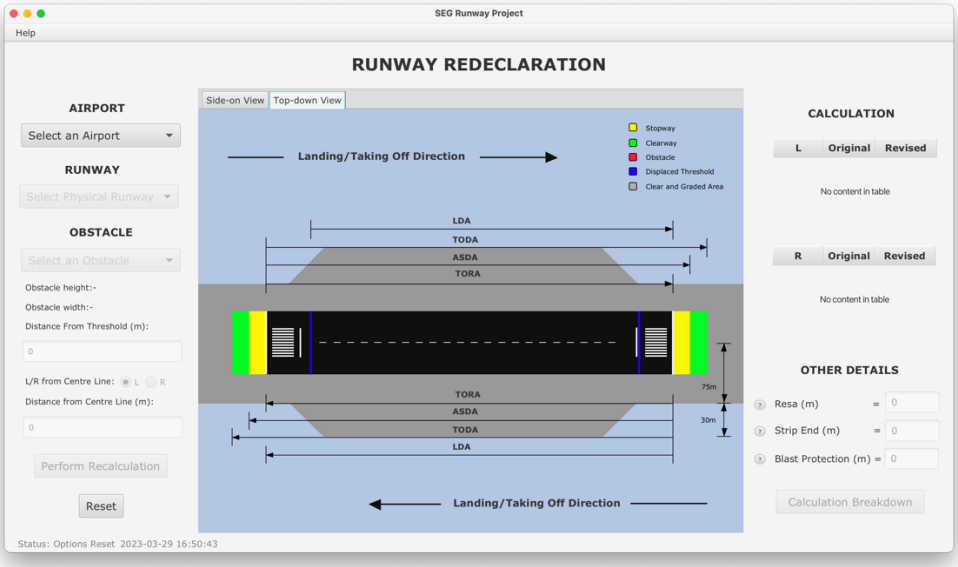
The current product delivery focuses on enhancing the system's visualisation capabilities for runway calculations. In the previous increment, the system displayed textual representations of information and result for redeclaration. However, in this deliverable, the group has added two visualisations, namely the side-on view and top view, which allow users to see changes to the runway parameters directly. The addition of the side-on view and top view visualizations enhances the user experience allowing them to easily view changes to the runway parameters and understand the effect of the changes on the overall visual representation of the runway shown on the system.

Furthermore, the GUI has undergone significant improvements and is now almost fully functional with most of the basic customer's requirements being addressed. For the next increment, the group plans to continue prioritizing user feedbacks to improve the system and enhance its functionalities based on the user requirements.

1.2 Application Details

Screenshots below show the entry page for our prototype, with the first one showing the side view visualisation and the second one showing the top-down visualization. There will be a leftmost pane for user input, and the rest of the area will be in the initial state without any calculation or details before user has keyed in the required details.

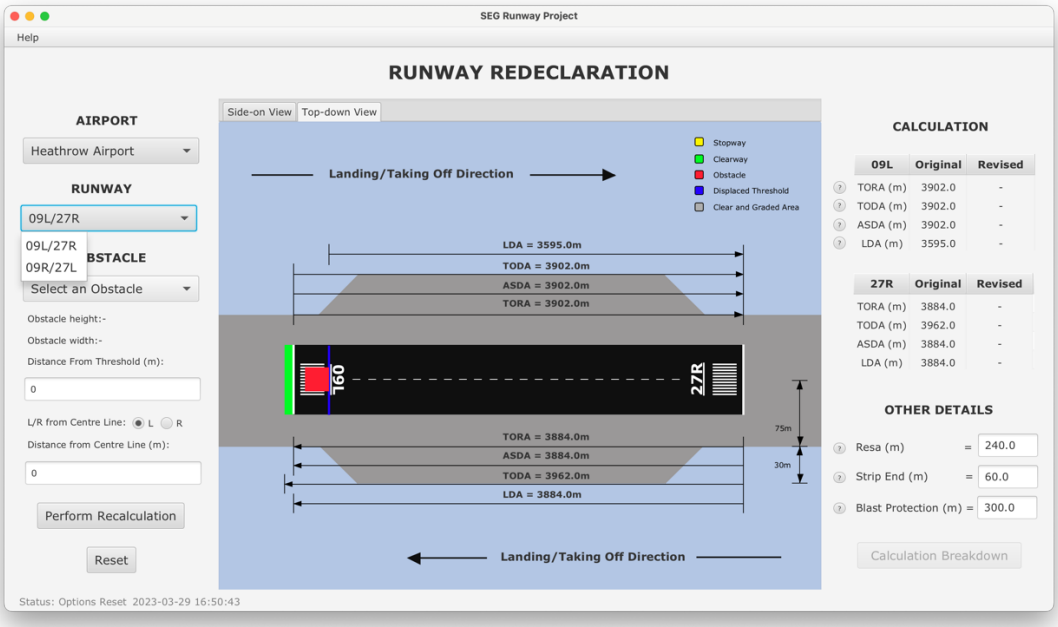




1.2.1 User Input

9	As an air traffic controller. I want to select different runways and thresholds, with views changing accordingly. So that the application is compatible with different runways.	Should

As shown in the screenshot below, user is able to select different runways and thresholds. The views will update according to the selection.

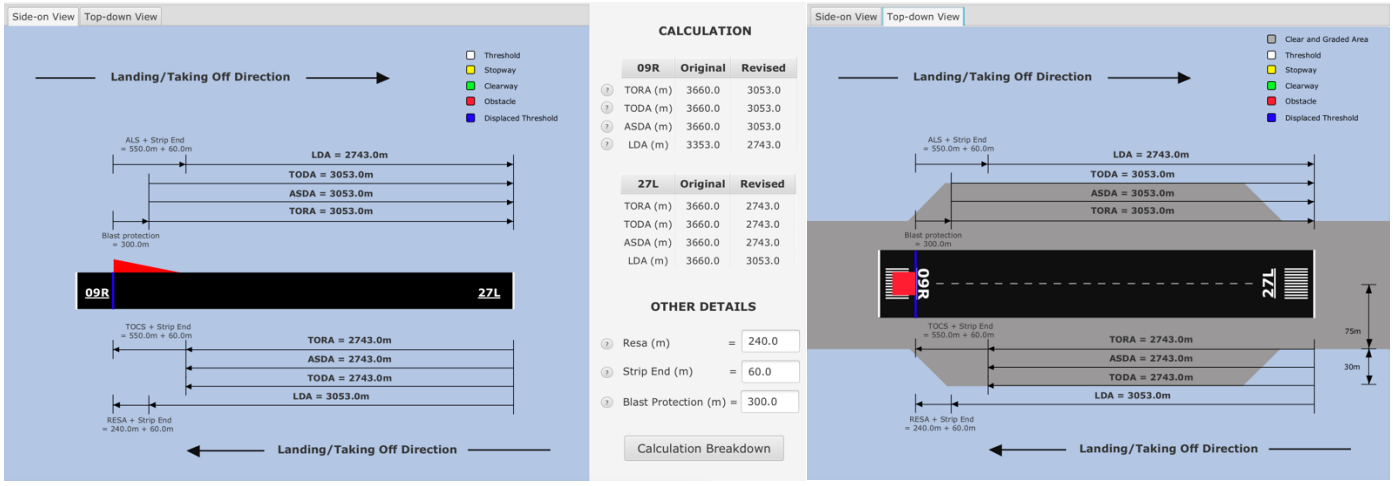


1.2.2 Visualisations

3	As an air traffic controller. I want to view the airport from top-down and side-on views simultaneously or individually in 2D. So that I can better visualize the runway.	Must

5	As an air traffic controller. I want to view the details of the runway and the distances used for runway declaration from both views. So that I can have a better understanding of how the runway declaration was done in the system.	Must

As seen from the screenshots below, when runway redeclaration is performed, the visualisations are being updated with the revised parameters.



4	As an air traffic controller. I want to view the cleared and graded areas in top-down view. So that I can manage the area more effectively.	Must

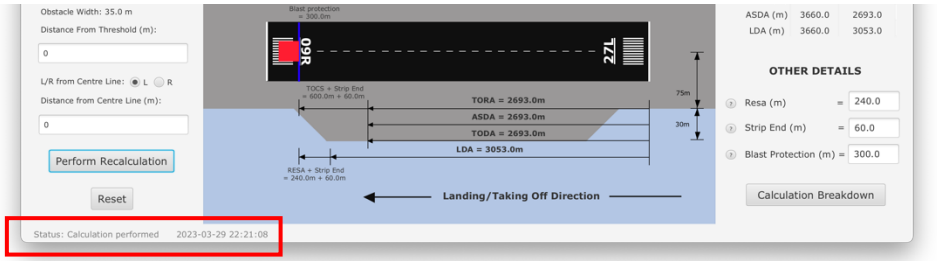
Referring to the screenshots above, the cleared and graded areas can be observed in the top-down view, which is colour coded as grey.

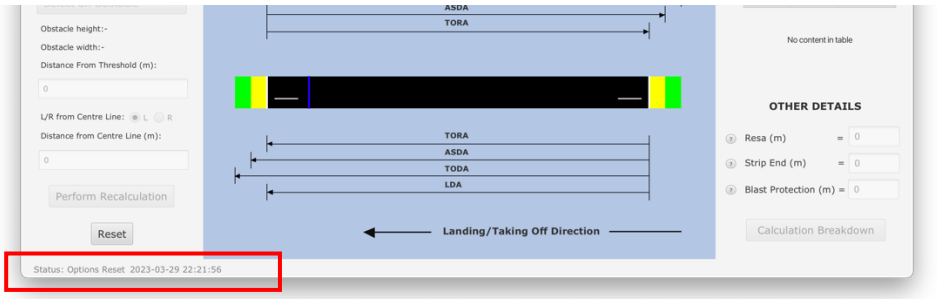
10	As an air traffic controller. I want to view the TOCS, and ALS slope caused by the obstacles in side-on view. So that I can make a better-informed decision.	Should

Similarly with the TOCS/ALS slope, it can be seen as the red polygon in the side-on view, indicating the slope caused by the obstacles.

14	As an air traffic controller. I want to have notifications displayed for any actions that occur. So that I am fully aware of what has taken place.	Could

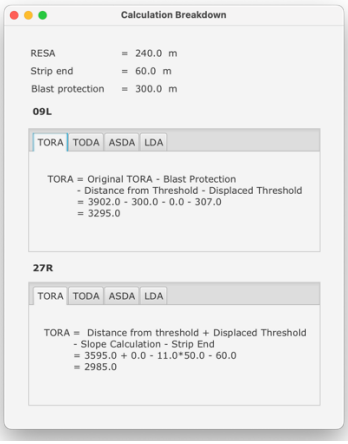
For this increment, there is only two main actions for the system that we think needs to be notified which is, whenever user perform a calculation and second is when user reset the system. The notification is implemented as a status bar at the lower part of the interface. It is shown in the form of action taken and a corresponding timestamp. Since we have decided to implement addition and editing of the runways and obstacles to the next increment, there is not much notifications needed at this stage. We have keep things simple for now but we do plan to extend this feature further in the next increment.



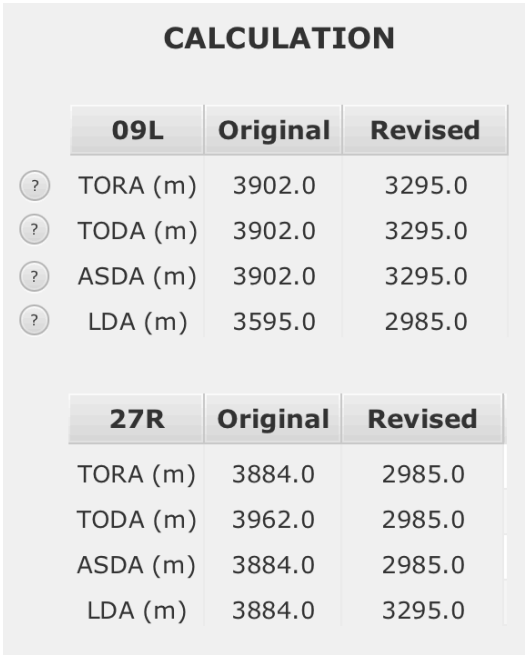


1.2.3 Other features/changes additional to the previous increment

- We have allocated the middle part of the interface for visualisation. The calculation breakdown is hidden and shown only if user click on the calculation breakdown button.



- The layout of the rightmost pane had been modified, the calculation result is now being shown in table form so that it is easier to compared original and revised values.



CALCULATION

09L	Original	Revised
TORA (m)	3902.0	3295.0
TODA (m)	3902.0	3295.0
ASDA (m)	3902.0	3295.0
LDA (m)	3595.0	2985.0

27R	Original	Revised
TORA (m)	3884.0	2985.0
TODA (m)	3962.0	2985.0
ASDA (m)	3884.0	2985.0
LDA (m)	3884.0	3295.0

- We have disable the logical runway selection in this increment. In the last increment, we have overlooked some details in the definition. As a result, we are only revising parameters in single direction. For example, when user select physical runway 09L/27R, they need to choose either 09L or 27R to perform the calculation. However, based on the definition, the recalculation need to be done in both directions and the requirement to always put lower value at the left makes more sense. So currently, user needs to only select a physical runway and recalculation will be carried out in both directions.

2 Design

2.1 Design Choices

2.1.1 Model-View-Controller Architecture

For this increment, we are still utilising the Model-View-Controller architecture and modifying the structure slightly to enable visualisations to be included.

2.2 Design Artifacts

In this increment, we are still using Unified Modelling Language (UML), a modelling tool that was used for planning and designing the general structure of our application. We decided to include use case diagram, sequence diagram, class diagram, and storyboard as key design artifact alongside with a user scenario.

2.2.1 User Scenarios

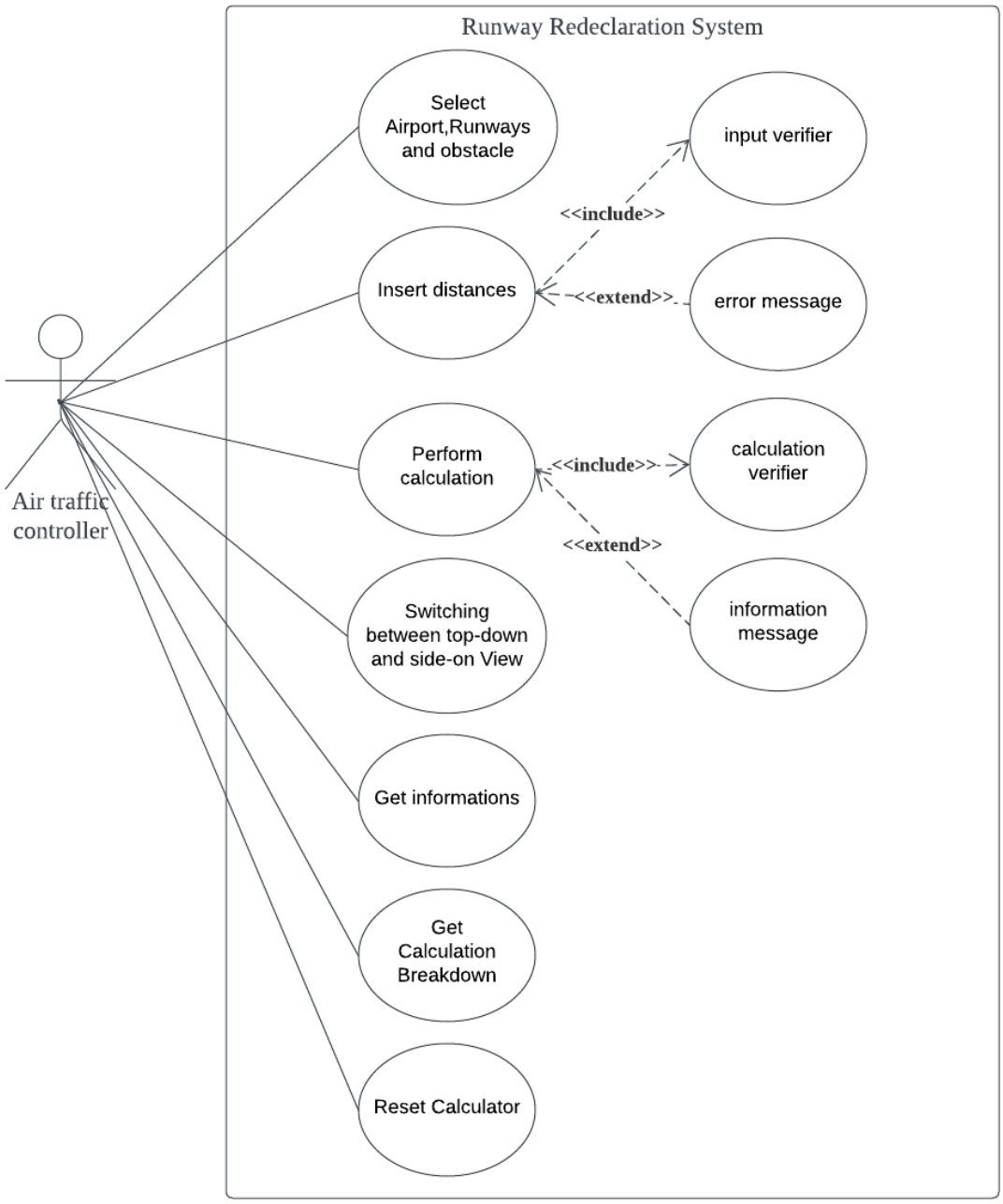
User scenarios were introduced during the development phase so that our team would have a better understanding on how users interact with the system.

2.2.1.1 Scenario 1

During peak hour, an airplane is trying to take-off while another airplane was blocking the runway. Due to the obstruction, the air traffic controller responsible the runway wants to have a quick redeclaration of the runway to see if the actual process is worthwhile. He also wanted to have a breakdown calculation to see if the calculations were correct and some visual to help him visualize the situation.

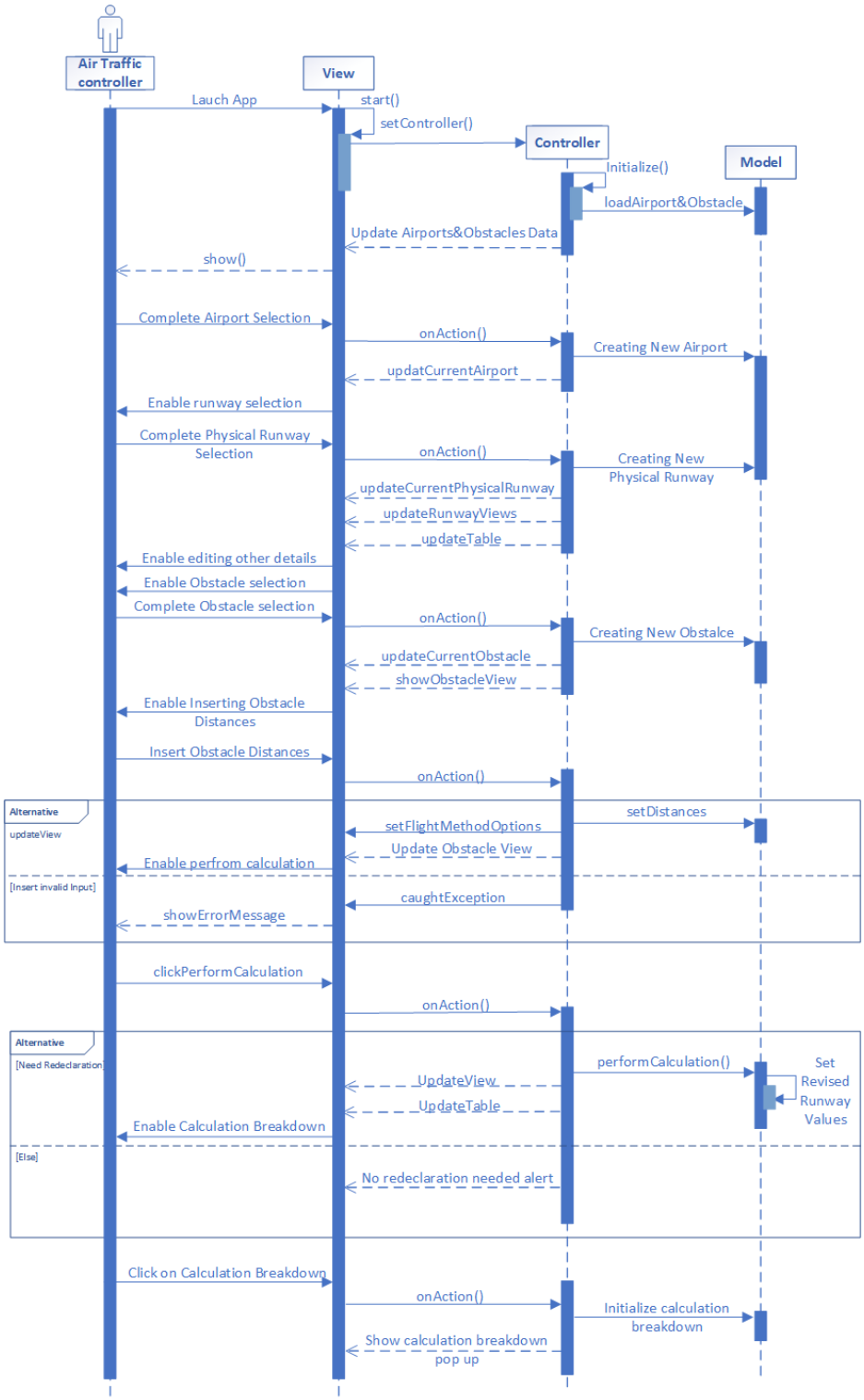
2.2.2 Use Case Diagram

This use case diagram was built based on the scenario above. The main focus of this diagram is to visualise how air traffic controllers interact with the system and help the user to understand what the development team will come up with in the early stage of development to clear up any misunderstanding between them.
(old use case as reference, update with the new one)



2.2.3 Sequence Diagram

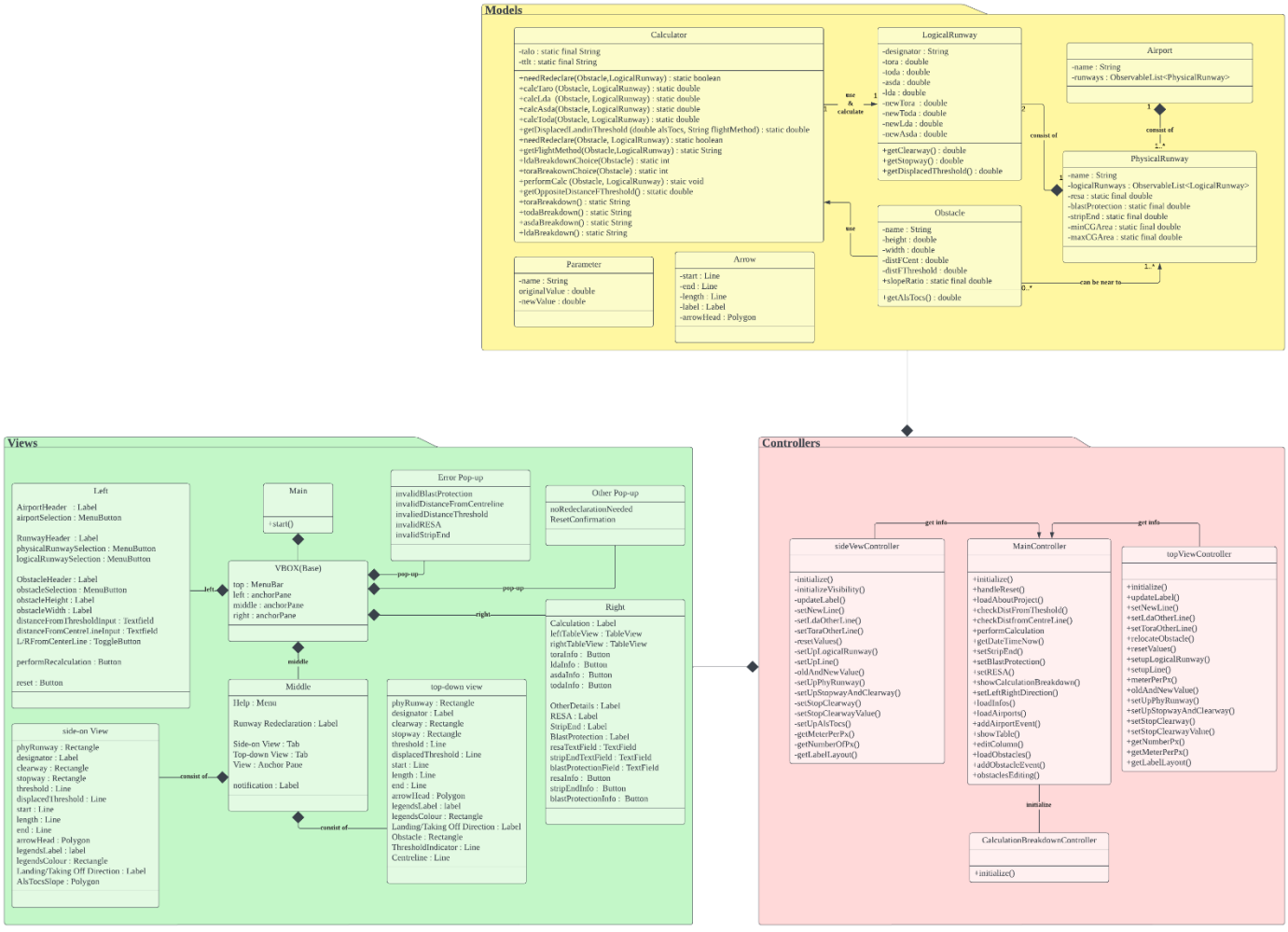
This sequence diagram illustrates how the actor (air traffic controller) interacts with a different part of the system that was based on MVC architecture. It also shows the flow of interactions between system when a task is performed by the user. In this scenario, the user is trying to redeclare runway declaration and access the calculation breakdown feature to compare his own paperwork to the system.



2.2.4 Class Diagram

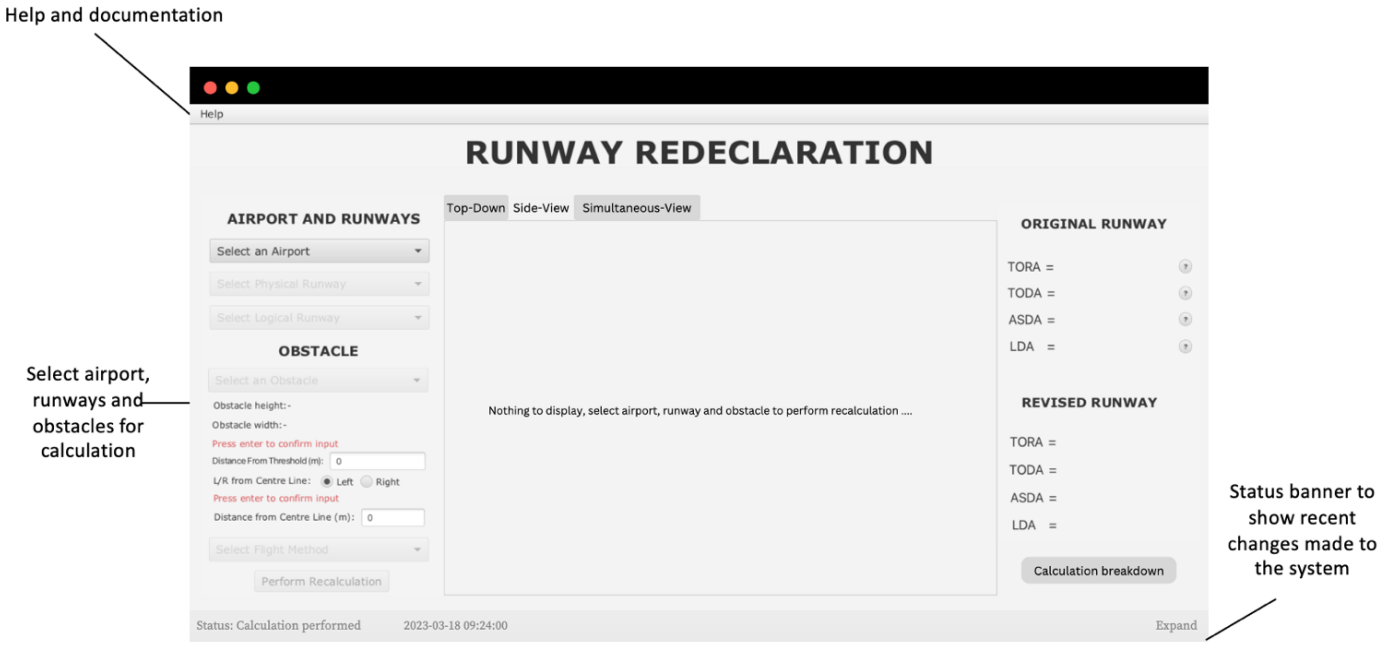
A class diagram was created to provide a clear understanding of the system structure among all the team members. This allows us to further divide our tasks in more detail while also working towards the same goal even when we are not physically working together all the time.

We are using the Model View Controller framework (MVC) to design the application. This class diagram consists of 3 different packages, which are, Models, Views, and Controllers. Each of them represents a different part of the system which has its functionalities respectively. Models is the backbone of the system which consist of object classes such as Airport, Physical Runway and Logical Runway. Views consist of mainly the user interfaces and some pop-up. Finally, controllers act as an interface between Models and Views, help updating Views while getting data from Models.

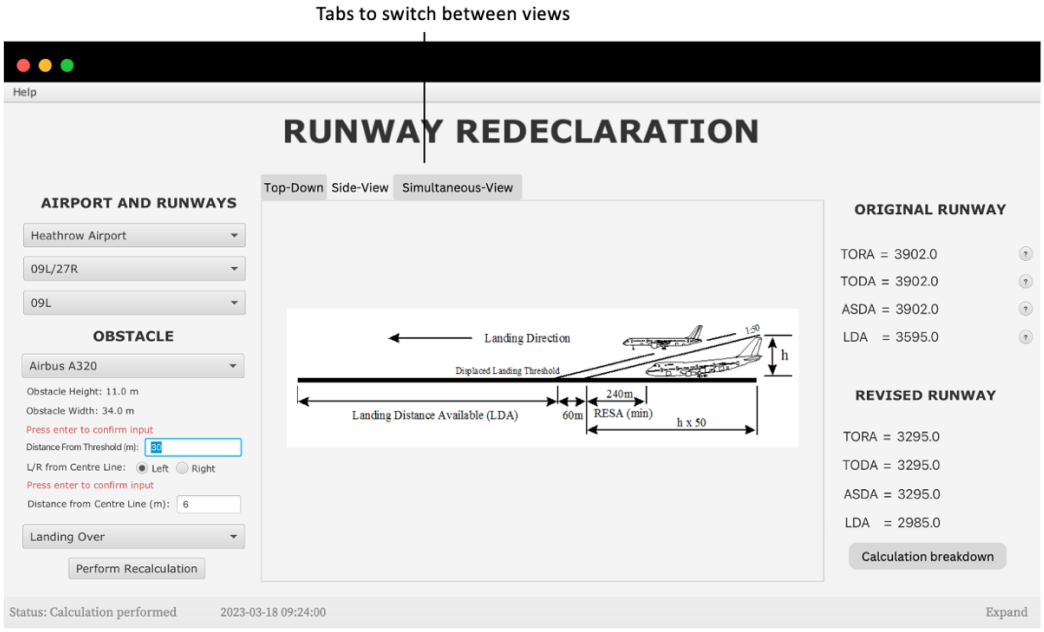


2.2.5 Storyboards

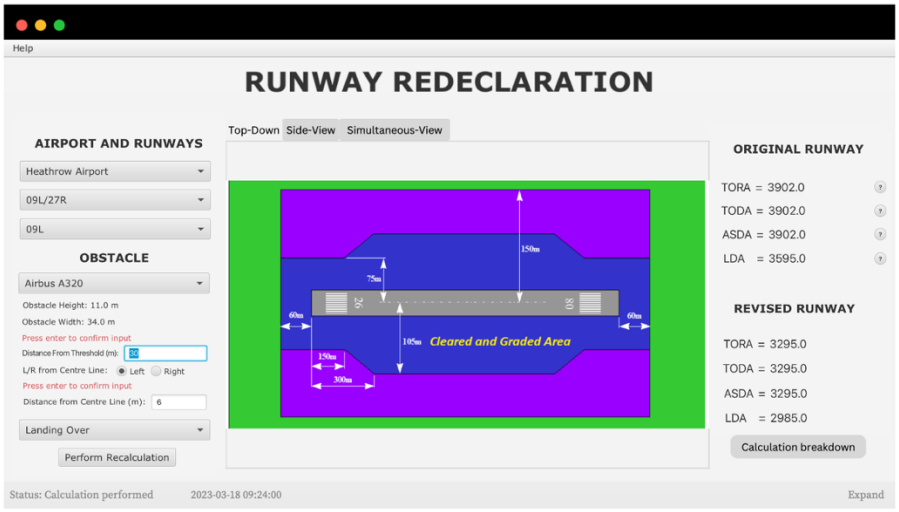
Initial version of storyboards of the system which includes five pages displaying the essential interfaces we intend to incorporate in this increment. The storyboards focus on some additional features to the last increment, which is the visualisation and slight adjustment of layout by moving calculation breakdown to a pop-up window that can be accessed by clicking on the button.



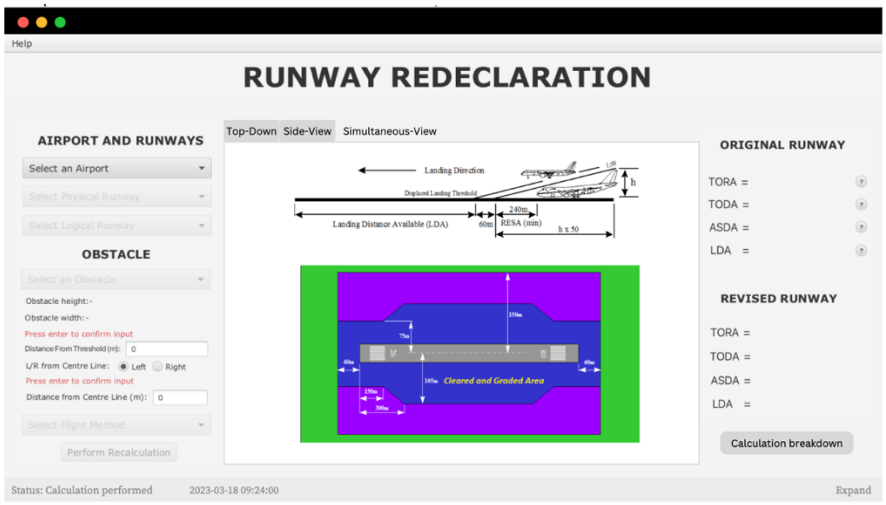
1. Starting page



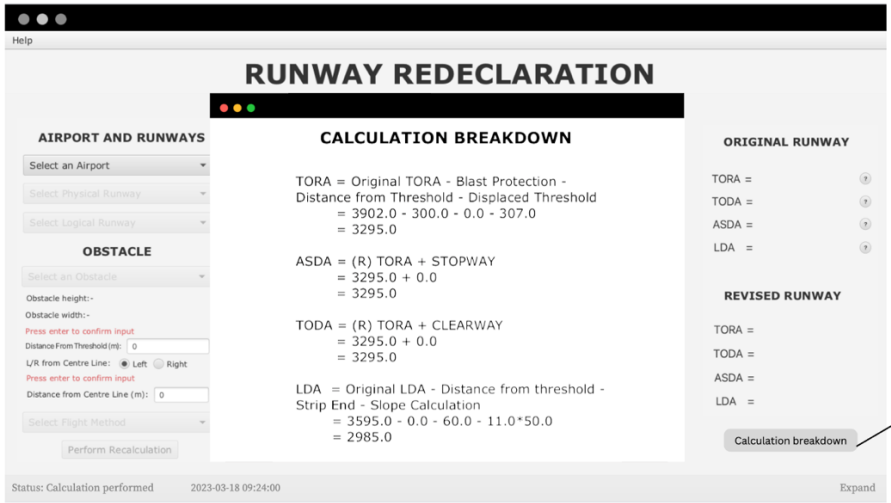
2. Side View Display



3. Top-Down View Display



4. Simultaneous-View Display



User can click on calculation breakdown to view details of the calculation

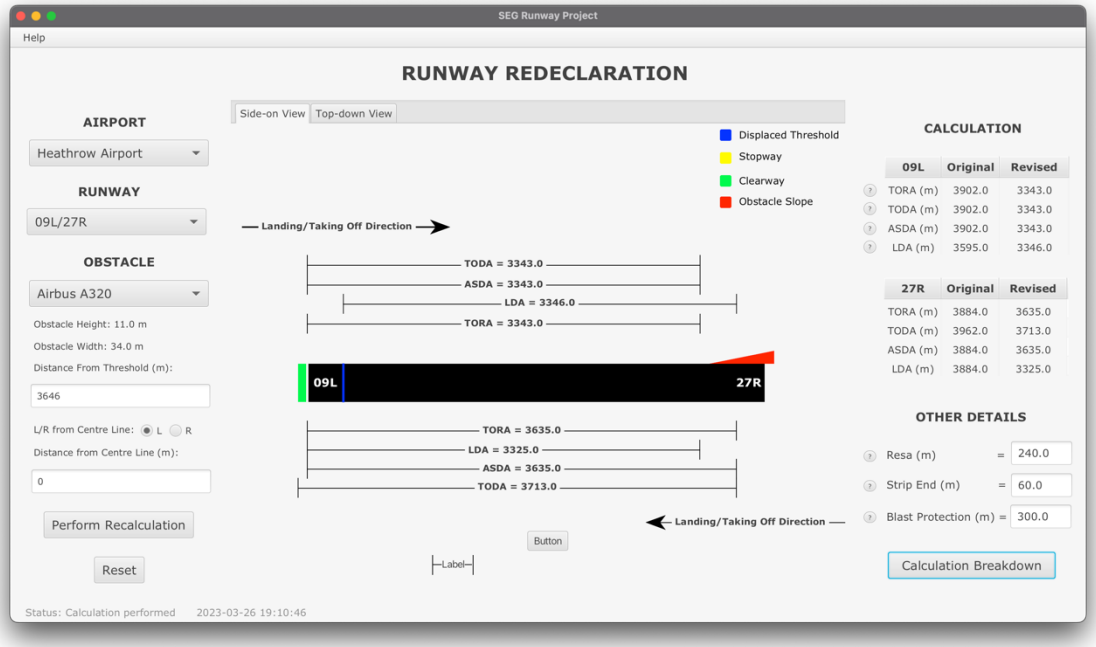
5. Calculation Breakdown Pop-up

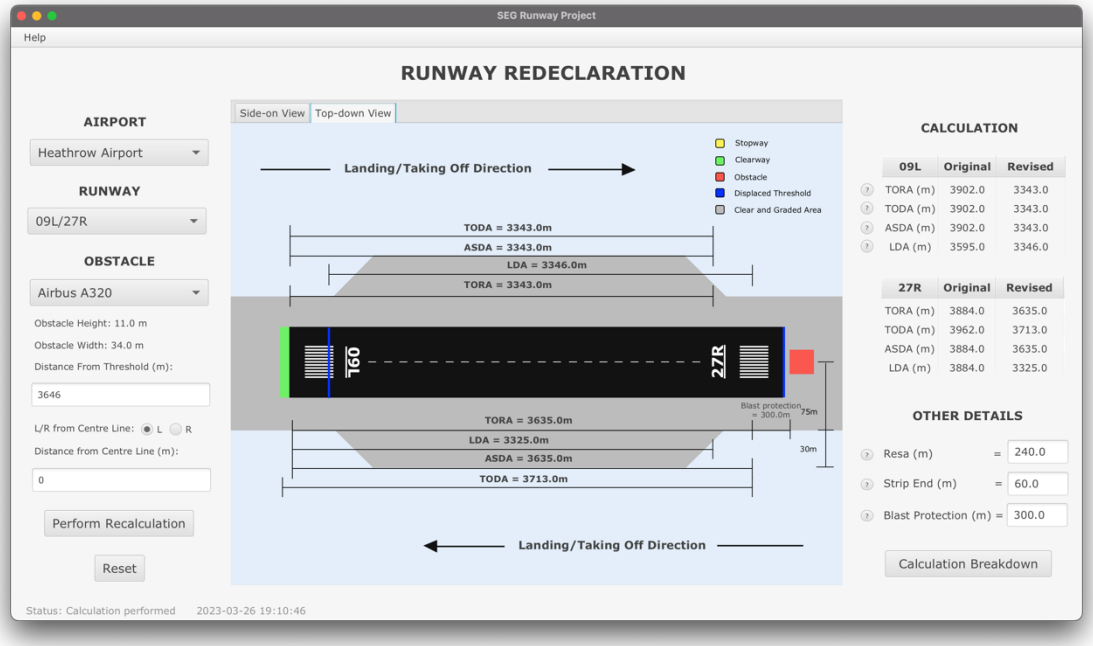
3 Product Testing

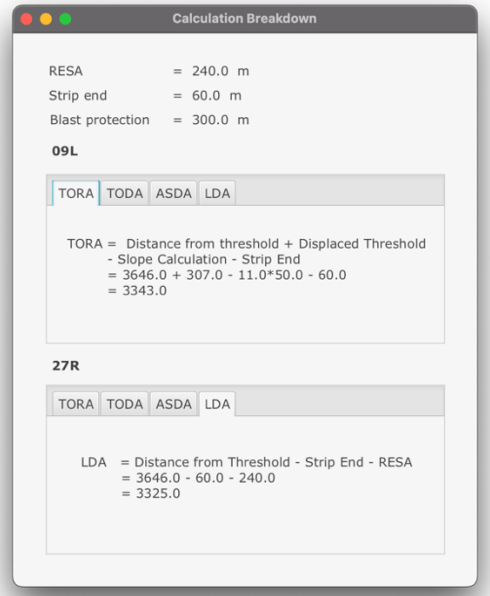
3.1 Test results

3.1.1 Success scenarios

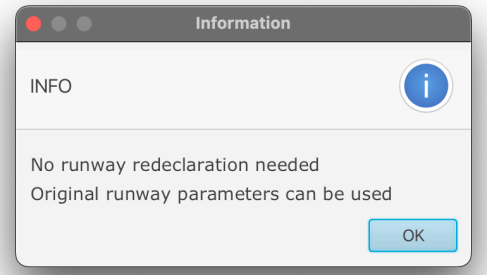
- The value of runway parameters is synchronised perfectly with the calculation breakdown:



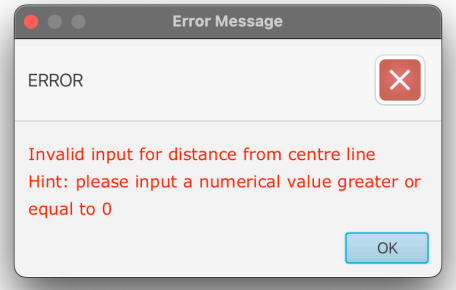




- For extreme values of input like 100000, for distance from threshold, it shows “No runway redeclaration needed, original runway parameters can be used”.

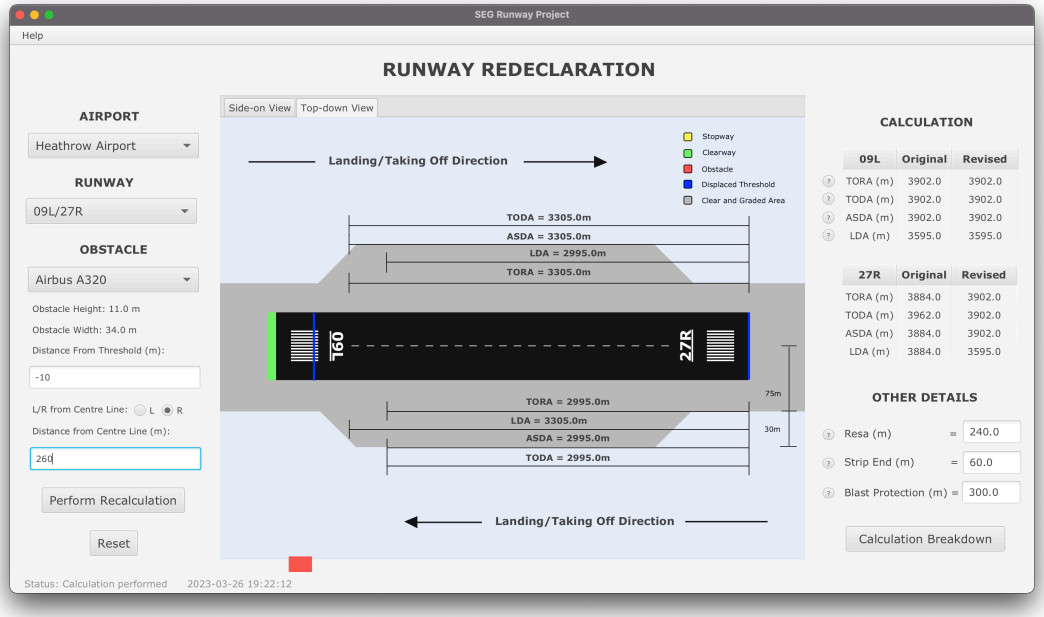


- For distance from center line, if value inputted is negative it shows “Invalid input for distance from center line Hint: please input a numerical value greater or equal to 0“.



3.1.2 Failed scenarios

- **Scenario 1:** For distance of centerline of around 260, the obstacle is beyond blue canvas



Problem identification:

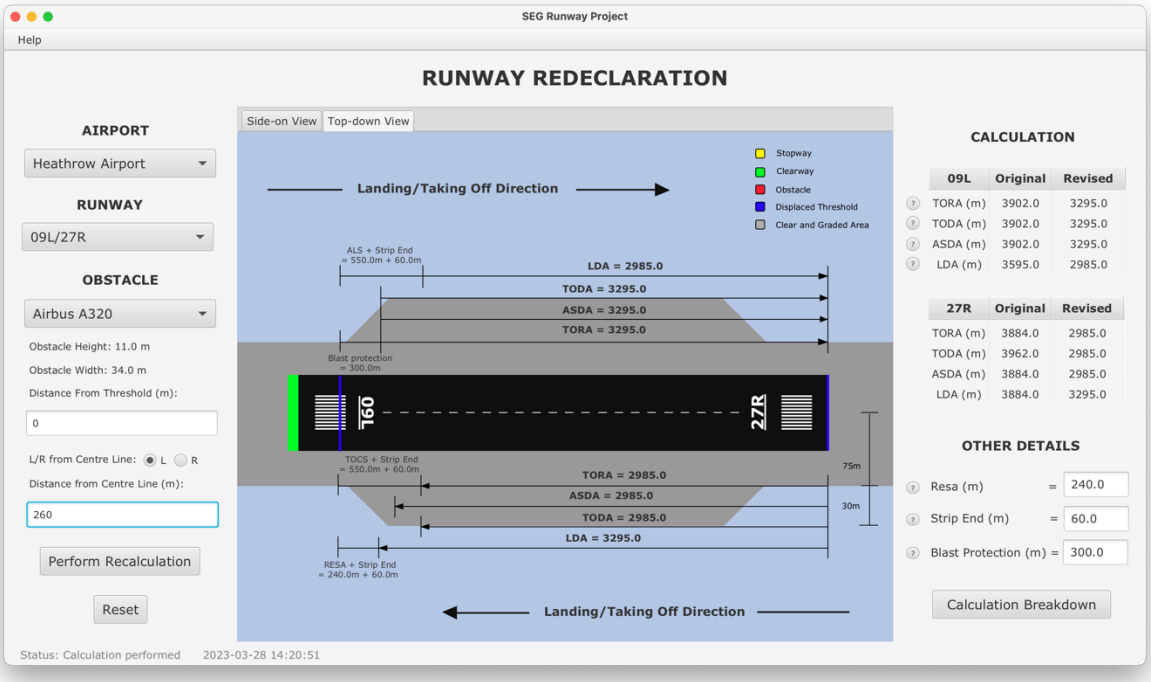
The problem with this is with the function that is used to relocate the obstacle and dimensions of the top view pane. The obstacle will be relocated even though no runway redeclaration is needed. The height of the top view pane is incorrect and smaller than the actual tab pane size and thus, with extreme values like 260, the obstacle is still visible but shown beyond the blue canvas.

Solution:

We have decided to not show the obstacle if there is no runway redeclaration needed (which means the obstacle is out of the area of runway that affects runway parameters).

```
534         double tora;
535         double stripEnd;
536         double obsBlockWidth = obstacleBlock.getHeight();
537
538         logRunway = MainController.getPhysRunwaySelected().getLogicalRunways().get(0);
539         tora = logRunway.getTora();
540         double displacedFromCentre = obstacle.getDirFromCentre().equals("L")? (-obstacle.getDistFCent()*(minCGArea.getHeight()/2)/
541                 obsBlockWidth/2: (obstacle.getDistFCent()*(minCGArea.getHeight()/2)/PhysicalRunway.minCGArea)-obsBlockWidth/2;
542         if(Calculator.needRedeclare(obstacle, logRunway)){
543             if(Calculator.getFlightMethod(obstacle, logRunway).equals("Take-Off Away Landing Over")){
544                 obstacleBlock.relocate( v: runwayStartX+((disFromThreshold+logRunway.getDisplacedThreshold())*(runwayLength-logRunw
545                         v1: centre+displacedFromCentre);
546             } else{
547                 obstacleBlock.relocate( v: runwayStartX+((disFromThreshold+logRunway.getDisplacedThreshold())*(runwayLength-logRunw
548                         v1: centre+(displacedFromCentre));
549             }
550         }else{
551             obstacleBlock.setVisible(false);
552         }
553     }
```

So, with the same selection and input for the centre line displacement, the obstacle is now not shown on the visualisation.



- **Scenario 2:** For both views, there is an issue with position of obstacle when different obstacle is selected. For example, when an obstacle A is selected and user input a distance from threshold of 300 m, if user made a different selection and select obstacle B instead, the obstacle is placed on threshold (with distance of threshold of 0 m instead of the previously inputted 300 m). However, if user click on the button to perform calculation, the position is updated correctly but with some inconsistencies.

Problem identification:

The problem lies in the action taken when different obstacle is being selected (listener in Main Controller).

Firstly, when a new obstacle is selected (obstacle B), the old obstacle's (obstacle A) distance from threshold is not being reset to the default value of 0. This causes obstacle A to be placed at the last inputted distance from threshold before selection changed (300 m in the above scenarios). To solve the issue, we have added lines of code to reset the distance from threshold to 0 in the listener in Main Controller.

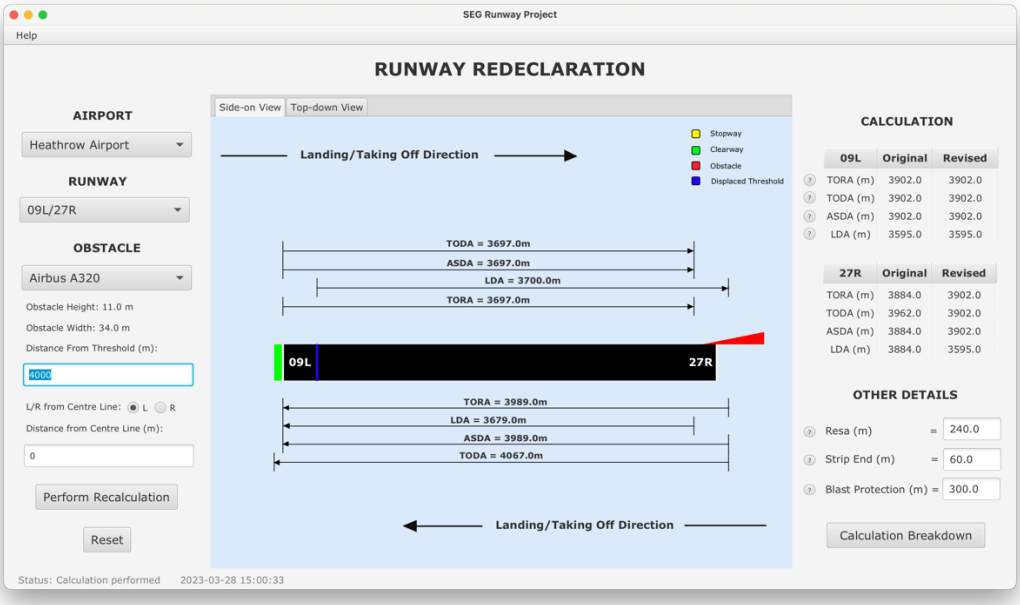
```
MainController.obstacleProperty.addListener((observable, oldValue, newValue) -> {
    if(oldValue != null){
        oldValue.setDistFThreshold(0);
        oldValue.setDistFCent(0);
    }
    newValue.setDistFThreshold(MainController.disFromThreshold.get());
    relocateObstacle();
});
```

Secondly, when new obstacle (obstacle B) is selected, it will have the default distance of 0 m from threshold, ignoring the previous input in the text field. To solve this, we added a line of code to set the distance from threshold for the new obstacle in the listener.

```
MainController.obstacleProperty.addListener((observable, oldValue, newValue) -> {
    if(oldValue != null){
        oldValue.setDistFThreshold(0);
        oldValue.setDistFCent(0);
    }
    newValue.setDistFThreshold(MainController.disFromThreshold.get());
    relocateObstacle();
});
```

Now the issue is solved with obstacle being positioned correctly before or after changes in selection.

- **Scenario 3:** For both views, if no redeclaration is needed the parameters are still recalculated and the view is being updated with new parameters and causing the labelling on the visualisation to be different from the result in tables.



Problem identification:

The problem is in listener in Main Controller, when perform calculation button is clicked on, both the side view controller and top view controller did not check if recalculation is needed but straightaway perform calculations and thus producing inconsistent results.

Solution:

We modify the function that handles label updating. We added a condition to check if recalculation is needed, if none is required, simply reset the parameter labels to original values. The original codes are being used if redeclaration is needed.

```
public void updateLabel(){
    relocateObstacle();
    PhysicalRunway selectedPhyRunway = MainController.getPhysRunwaySelected();
    LogicalRunway llogRunway = selectedPhyRunway.getLogicalRunways().get(0);
    Obstacle obstacle = MainController.getObstacleSelected();
    if(Calculator.needRedeclare(obstacle, llogRunway)){
        Calculator.performCalc(obstacle,selectedPhyRunway);
        resetValues(selectedPhyRunway);

        setNewLine( type: "TORA", LeftorRight: "Left",selectedPhyRunway,obstacle,toraStart,toraLength,toraEnd,toraLabel,toraArrow);
        setNewLine( type: "LDA", LeftorRight: "Left",selectedPhyRunway,obstacle,ldaStart,ldaLength,ldaEnd,ldaLabel,ldaArrow);
        setNewLine( type: "ASDA", LeftorRight: "Left",selectedPhyRunway,obstacle,asdaStart,asdaLength,asdaEnd,asdaLabel,asdaArrow);
        setNewLine( type: "TODA", LeftorRight: "Left",selectedPhyRunway,obstacle,todaStart,todaLength,todaEnd,todaLabel,todaArrow);
        setNewLine( type: "TORA", LeftorRight: "Right",selectedPhyRunway,obstacle,toraEnd1,toraLength1,toraStart1,toraLabel1,toraArrow1);
        setNewLine( type: "LDA", LeftorRight: "Right",selectedPhyRunway,obstacle,ldaEnd1,ldaLength1,ldaStart1,ldaLabel1,ldaArrow1);
        setNewLine( type: "ASDA", LeftorRight: "Right",selectedPhyRunway,obstacle,asdaEnd1,asdaLength1,asdaStart1,asdaLabel1,asdaArrow1);
        setNewLine( type: "TODA", LeftorRight: "Right",selectedPhyRunway,obstacle,todaEnd1,todaLength1,todaStart1,todaLabel1,todaArrow1);

        setToraOtherLine(Calculator.needRedeclare(obstacle, llogRunway), Calculator.getFlightMethod(obstacle, llogRunway).equals(Calcul
        setLdaOtherLine(Calculator.needRedeclare(obstacle, llogRunway), Calculator.getFlightMethod(obstacle, llogRunway).equals(Calcula

        setToraOtherLine(Calculator.needRedeclare(obstacle, llogRunway), Calculator.getFlightMethod(obstacle, llogRunway).equals(Calcul
        setLdaOtherLine(Calculator.needRedeclare(obstacle, llogRunway), Calculator.getFlightMethod(obstacle, llogRunway).equals(Calcula
    } else{
        resetValues(selectedPhyRunway);
        setUpLogicalRunway(selectedPhyRunway);
    }
```

- **Scenario 4:** The program is able to start in Intellij but when exported as .jar file, fxml files including SideView.fxml, TopView.fxml and CalculationBreakdown.fxml failed to load giving an error of nullPointerException.

```
Run:  RunwayRedeclaration.jar
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java --module-path /Users/jingru/javafx-sdk-19.0.2.1/lib --add-modules=ja
java.lang.NullPointerException Create breakpoint
    at java.base/java.util.Objects.requireNonNull(Objects.java:233)
    at Controller.MainController.initialize(MainController.java:152)
    at javafx.fxml@19.0.2.1/javafx.fxml.FXMLLoader.loadImpl(FXMLLoader.java:2662)
    at javafx.fxml@19.0.2.1/javafx.fxml.FXMLLoader.loadImpl(FXMLLoader.java:2555)
    at javafx.fxml@19.0.2.1/javafx.fxml.FXMLLoader.load(FXMLLoader.java:2523)
    at View.Main.start(Main.java:30)
    at javafx.graphics@19.0.2.1/com.sun.javafx.application.LauncherImpl.lambda$launchApplication1$9(LauncherImpl.java:847)
    at javafx.graphics@19.0.2.1/com.sun.javafx.application.PlatformImpl.lambda$runAndWait$12(PlatformImpl.java:484)
    at javafx.graphics@19.0.2.1/com.sun.javafx.application.PlatformImpl.lambda$runLater$10(PlatformImpl.java:457) <1 internal line>
    at javafx.graphics@19.0.2.1/com.sun.javafx.application.PlatformImpl.lambda$runLater$11(PlatformImpl.java:456)
    at javafx.graphics@19.0.2.1/com.sun.glass.ui.InvokeLaterDispatcher$Future.run(InvokeLaterDispatcher.java:96)
```

Problem identification:

The issue lies in path for the fxml files, running Main.java works well in Intellij because path to all the fxml files are correct and relative to Main.java. However, the .jar file is build outside of src folder and thus it cannot access those fxml files with the same relative paths.

Solution:

By referring to online forums, specifically this one, https://stackoverflow.com/questions/17228487/javafx-location-is-not-set-error-message, one user managed to solve the problem by utilising a resource folder and putting fxml files under them. This way, we can access fxml files directly by putting a '/' before the file name. Based on this, https://stackoverflow.com/questions/17228487/javafx-location-is-not-set-error-message we were able to create a resource folder to keep all the fxml files.

The .jar file is able to run without error now.

Note: Testing is carried out during the development process so some of the interface screenshots are different from the final interface.

4 Second Sprint Review

The following paragraphs are the key points of our sprint review. It consists of the successes we had during this sprint as well as challenges, which the team will seek to improve during the following sprint.

4.1 Successes and Challenges

Successes

Our team concentrated mostly on the GUI and runway display for this sprint because the majority of the calculations had already been implemented in the previous increment. Also, we had strong foundations because core features, functionalities and unit testing had already been completed. As a result, the team's attention was completely on the visualisation/GUI aspect and was not diverted to the problems with the previous increment. Hence, in this iteration, the team was successful in creating a fully functioning, user-friendly, and intuitive graphical user interface that satisfies key user requirements.

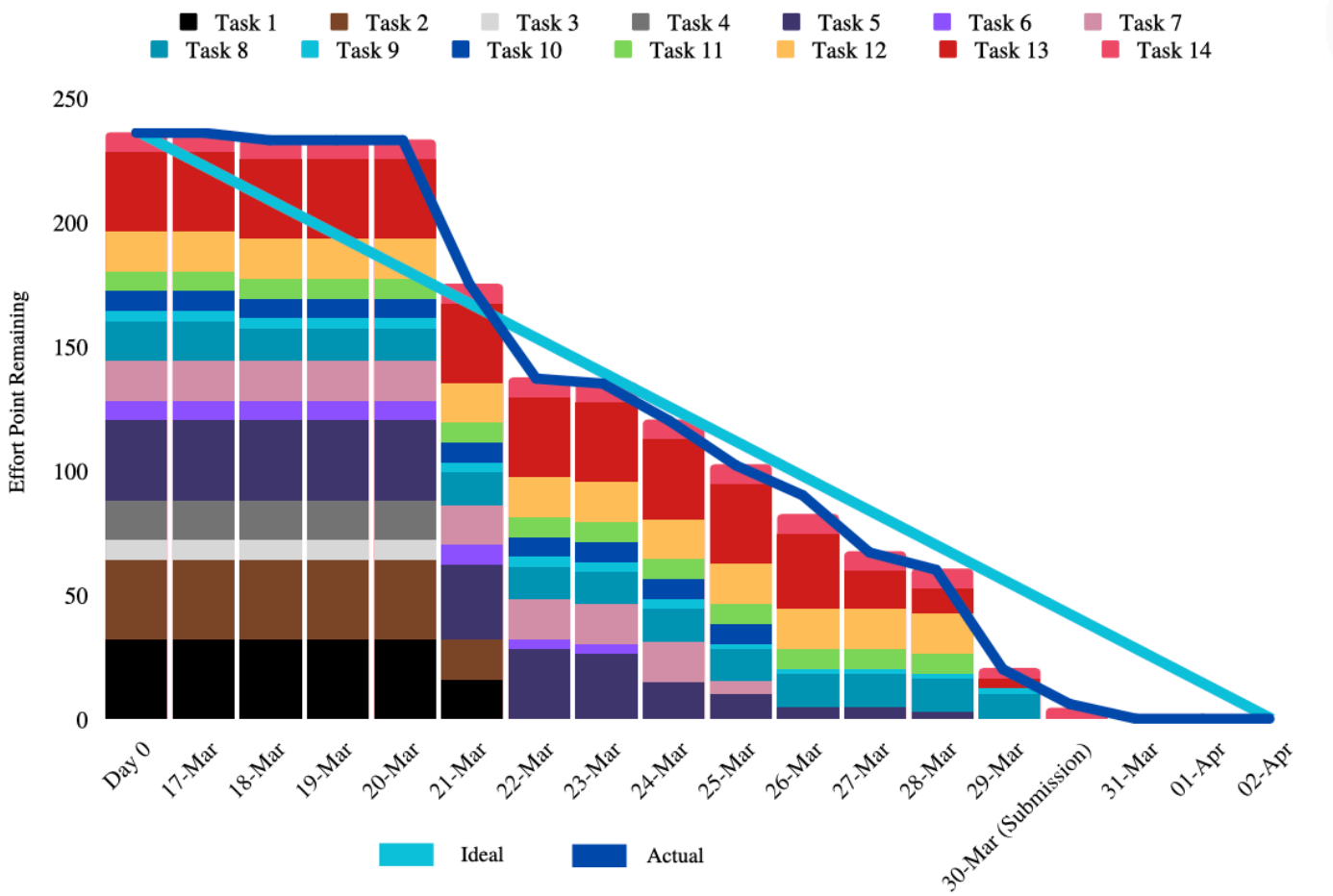
Challenges and Solutions

Despite the fact that our team was successful in building a fully functional calculator, there were some challenges. Firstly, because of the study break we weren't able to have two meetings with our supervisor and users, preventing the team from getting their input and feedback on our progress at the time. Nonetheless, the usage of WhatsApp/Discord allowed the team members to communicate with one another frequently. Lastly, there were several changes made to the GUI/visualizations, which led to errors that affected the previous codes thus the team had to spend additional time adjusting the code accordingly.

4.2 Sprint Burn-down Chart

Table 1.1 Tasks for Second Sprint: List of tasks for second sprint with effort points determined using planning poker

No.	Tasks	Effort (pts)
	Application - Functionalities	
1.	Create a basic layout for side view	32
2.	Create a basic layout for top-down view	32
3.	Clear display for cleared and graded area	8
4.	Clear representation of new runway parameters in the visualizations	16
5.	Views changing dynamically based on runways selected and other inputs	32
6.	TOCS and ALS slope caused by obstacles being shown clearly in side-view	8
7.	Notification feature to keep track of changes to the system	16
	Planning and Report Writing	
8.	Using UML diagrams, storyboards to support design decisions	16
9.	Construction of burndown chart for second increment based on actual progress	4
10.	Introduction of sprint plan for third increment	8
11.	Response to feedback	8
12.	Application demo and screenshots	16
	Testing	
13.	Test and verifying the correctness of application	32
	Submission	
14.	Finalizing submission folder, exporting .jar file, writing readme	8


Task 1
Task 2
Task 3
Task 4
Task 5
Task 6
Task 7
Task 8
Task 9
Task 10
Task 11
Task 12
Task 13
Task 14
Effort Point Remaining
250
200
150
100
50
0
Day 0
17-Mar
18-Mar
19-Mar
20-Mar
21-Mar
22-Mar
23-Mar
24-Mar
25-Mar
26-Mar
27-Mar
28-Mar
29-Mar
30-Mar (Submission)
31-Mar
01-Apr
02-Apr
Ideal
Actual

5 Third Increment Planning

5.1 Sprint Plan for the Next Increment

As an air traffic controller. I want to import and export details of obstacles, airports, and other data using appropriate XML files So that they can be stored/transferred to others.
As an air traffic controller. I want to be able to use it at any UK commercial airport. So that I can access it at the control tower.
As an air traffic controller. I want to print out the situation of the runway in textual format. So that it could be revise with ease.

Extra backlog items that can be implemented if time permits. (Note: They will not be counted as effort points for this increment as these will be extensions to the systems)

As an air traffic controller. I want to automatically rotate the runway strip to match its compass heading. So that the direction of the runway is clear and easily identifiable.
As an air traffic controller. I want to zoom and pan the views. So that I can focus on the part of the runway that I am working with.

Table 1.2 Tasks for Third Sprint: List of tasks for third sprint with effort points determined using planning poker

No.	Tasks	Effort (pts)
	Application - Functionalities	
1.	Allow user to add new airports through GUI form or XML files	16
2.	Allow user to edit existing obstacle details (height and width)	16
3.	Allow user to export airport details in XML files	4
4.	Allow user to export obstacle details in XML files	4
5.	Improve on GUI design using CSS	32
6.	Implement login interface	32
7.	Add more information in help and documentation	16
8.	Automatically rotate runway strip to match compass heading	32
9.	Extend on notification features	16
	Planning and Report Writing	
10.	Using UML diagrams, storyboards to support design decisions	16
11.	Construction of burndown chart for third increment based on actual progress	4
12.	Response to feedback	4
13.	Application demo and screenshot	8
	Testing and Error Handling	
14.	Checking for invalid inputs (in airport XML)	16
	Submission	
15.	Finalizing submission folder, exporting .jar file, writing readme	4

5.2 Day-Zero Burndown Chart